3D mapping and Object Segmentation

Nishant Pandey
University of Maryland
npandey2@umd.edu

Jayasuriya Suresh
University of Maryland
jsuriya@umd.edu

Rishikesh Jadhav
University of Maryland
rjadhav1@umd.edu

Abstract

Indoor navigation for robots has become a crucial part of their use in such environments. Mapping an environment allows autonomous navigation to detect and avoid obstacles. In this paper, we present a novel approach that utilizes RGB images to segment indoor environments. The paper introduces a method for indoor mapping that can be used for robot navigation by first creating a 3D mesh from Multi-View Stereo (MVS) RGB images and then converting this mesh into a point cloud for environmental segmentation. We carry out experiments to establish a baseline for our method. We present our findings and provide avenues for future work.

1. Introduction

The domain of indoor mapping is a useful tool for creating real-world representations used in robot navigation, digital twins of scenes, and various other applications. These representations have diverse applications across sectors. For instance, in autonomous robotics, indoor mapping technology significantly improves the efficiency and safety of robots navigating complex spaces such as warehouses, hospitals, or retail stores. In emergency scenarios like firefighting or search-and-rescue operations, this technology offers detailed maps of uncharted or dangerous indoor areas, thereby enhancing safety and situational awareness. Furthermore, in virtual and augmented reality, indoor mapping is utilized to create precise and immersive digital replicas of real-world indoor environments, enriching experiences in gaming, training simulations, and virtual tours.

Creating 3D maps and segmenting objects within them are essential for robot navigation in indoor settings. Robots rely on this data to plan routes and avoid obstacles. Object segmentation is crucial as it helps robots navigate around these objects.

Recent progress in indoor mapping has harnessed various imaging and sensor technologies to produce highly accurate 3D scene mappings. Seichter *et al*. [19] use voxel-based representations that can perform competitively compared to point clouds but are limited by the resolution of 3D grids in the quality of their representations. Teng *et al*. [20] propose using a bird's eye view from a 360-degree FOV video of indoor scenes to do segmentation. Works like Sandström *et al*. [17] focus on point cloud representations for reconstructing and segmenting 3D scenes, primarily using RGBD input. Castagno *et al*. [1] present another approach.

Our research aims to achieve similar outcomes using RGB images, without additional visual inputs. We propose a pipeline that reliably segments scenes using a readily available RGB camera, ensuring efficient processing.

While a variety of sensors can be used to acquire environmental information, Lidar sensors and cameras are commonly used. We use an RGB camera as the method of acquiring data from the surroundings. RGB cameras are more affordable compared to Lidar sensors while being able to cover a larger field of view and gather more information. RGBD cameras provide an alternative but come with a higher cost and bulkier hardware that can result in more power draw, which is critical for battery-powered robots. The use of RGB-only images also allows for quicker processing of information gathered, crucial in real-time applications. We seek to combine the benefits of using RGB images along with the quick outputs of point cloud segmentation methods to achieve segmentation of a scene. The advantages of using images are also that they can be efficiently stored without significant loss of detail.

Our approach involves the combination of SimpleRecon [18] and Point-Voxel CNN (PVCNN) [10], which are quick, not memory-heavy, and produce good outputs. These characteristics make them suited for real-time applications.

Point Cloud representations of objects are easy to process and segment in real-time applications. However, they can miss topological features of surfaces and objects that can be crucial in integrating multiple scene viewpoints. Our proposed approach leverages mesh representations to overcome this, offering rapid and precise scene reconstructions. The use of meshes provides twofold benefits: quick and highly accurate reconstructions of scenes and the ability to densely or sparsely sample points from meshes, allowing

for flexibility and efficiency in processing point clouds for segmentation. These meshes, derived from images and their depth equivalents, are then transformed into point clouds. This method combines the speed of point cloud segmentation with the detailed accuracy of mesh representations.

To summarize our contributions:

- We propose a novel method for indoor mapping using an RGB camera that can generate a segmented output for navigational tasks.
- We conducted experiments to demonstrate the viability of our pipeline and visualize the results.
- We analyze and discuss our results. We also list improvements to enhance our results and avenues for future work.

2. Related Work

2.1. Indoor Scene Mapping

There have been many works in recent times that have done 3D mapping of environments utilizing various techniques. These tend to be an end-to-end pipeline from input to output. Many works tend to use point cloud representations obtained from monocular images, RGBD (Red, Green, Blue, and Depth) images, and lidar data. Seichter *et al*. [19] use voxel-based representation to carry out mapping and segmentation, which has been shown to give good results but is not very quick.

Sandström *et al*. [17] use point clouds for creating maps with high accuracy for the task of Visual SLAM but make use of an RGBD monocular camera. Chen Yang *et al*. [21] proposed a framework that uses monocular camera information to estimate camera poses and generates a point cloud representation of a scene to undergo SLAM with promising results. Karam *et al*. [6] make use of sensor fusion from a drone using lidar data and generate a map of a scene that can be easily updated and is efficient.

Castagno *et al*. [1] creates meshes from images that are high quality but use an RGBD camera to gather images with depth and use point clouds to generate meshes. The author also does not make use of any deep learning-based method, but a line of work could involve using the ideas proposed in Polylidar3D [1] to build a framework for segmentation.

Robert *et al*. [16] propose using a combination of 2D and 3D data input without requiring depth maps to map and segment 3D scenes, thus making it ideal for sensor fusion. It can also utilize multiple views of a scene along with lidar data. We drew inspiration from this work for the use of multi-view Stereo Images.

Zimmerman *et al*. [23] make use of RGB images along with floor plan priors to map out an indoor scene that can be used to keep an updated record of objects within the scene. This approach is not suitable for environments where many large objects may be frequently moved, like warehouses.

Sayed *et al*. [18] make use of MVS images, which are used to predict the depth and generate a mesh representation of an indoor scene and are efficient and quick. We incorporate this into our proposed pipeline.

2.2. Scene Schematic Segmentation

The fields of scene mapping and segmentation relate to each other very closely, with the use of a reconstruction of a scene often but not a necessity. As described under mapping, Robert *et al*. [16] do mapping, and the generated map is used to be segmented. Teng *et al*. [20] take input as a 360-Degree FOV video of indoor scenes to segment it, but this approach poses numerous challenges in extracting features from such videos.

Manglani *et al*. [12] utilize scene segmentation to navigate a robot in an indoor environment. The proposed method is simple and efficient; however, it is unable to make use of any 3D data.

2.3. PointCloud segmentation

There have many been neural network-based point cloud segmentation models proposed over the years. One of the popular ones is PoinetNet [14] and its derivatives [10, 15] that offer good performance with simple architectures. We use Point-Voxel CNN in our work as it uses voxel feature convolution along with point clouds for the segmentation of scenes. It is possible to extract the voxels and use them for further downstream processing if required for other applications. These can handle scene segmentation. The irregular nature of point clouds makes it hard to determine neighbor points quickly and can result in expensive computation time while using memory to store the information. Also, dynamic kernel computation of neighbors due to the unordered nature of point clouds results in a large overhead in the pipeline. The use of voxel-based additions reduces these overhangs and enhances good data locality and regularity. This combines the best of point clouds and voxel representation to achieve major time reductions with better performance.

Many other works make use of multi-modal inputs and different deep networks to improve the performance over Pointnet. Lai *et al*. [8] proposed the use of a transformer for point cloud segmentation with good accuracy. Liu *et al*. [9] makes use of image-language models to improve part segmentation but could be extended to scene segmentation. Krispel *et al*. [7] proposed FuseSeg, an extension of SqueezeSeg that makes use of multi-modal data, RGB data, and lidar data to do scene segmentation of outdoor scenes. This has highly promising results if it is extended to indoor segmentation.

3. Method

Our proposed method intends to do indoor scene reconstruction and carry out segmentation of the same. It is a

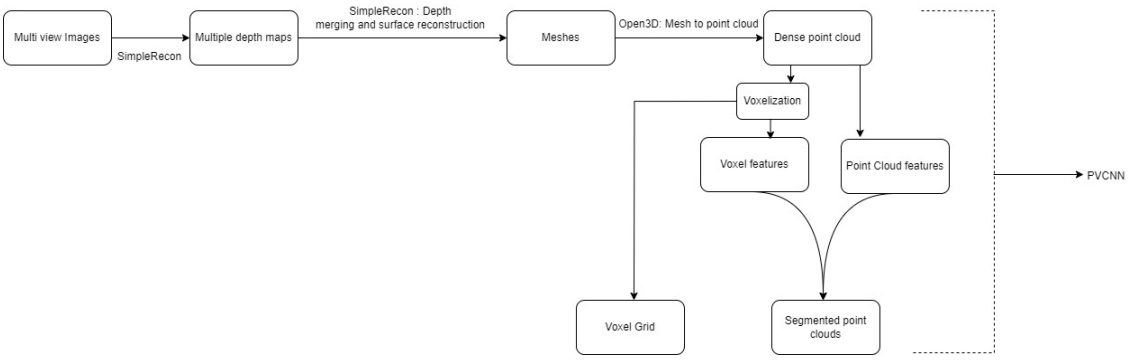


Figure 1. Execution Pipeline

novel pipeline that combines existing methods and executes them to get the desired outcome. During the entire course of our research, there were a few pipelines that we experimented on to get the desired results. We explain our current pipeline and the prior pipeline as a study of what was done to improve outputs. Our pipeline involves the following steps.

The pipeline diagram 1 above demonstrates our execution steps. The overview of the same is as follows:

1. Obtaining depth maps using a reference image, a set of images along with their intrinsic and relative camera poses. (SimpleRecon [18])
2. Depth merging and surface reconstruction (simple off-the-shelf depth fusion).
3. Converting mesh data to point cloud using vertex sampling method (Open3D).
4. Using Point-Voxel Convolution [10] for point cloud segmentation.

3.1. Depth maps

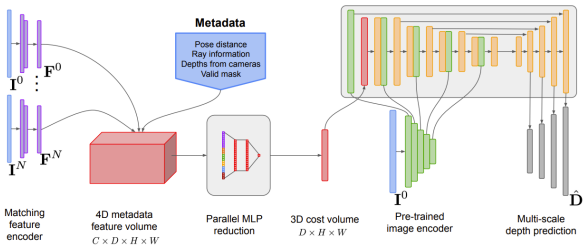


Figure 2. SimpleRecon architecture to obtain depth maps [18]

SimpleRecon enhances 3D indoor scene reconstruction by combining a 2D CNN with a cost volume for depth prediction. This approach innovatively integrates metadata into the cost volume, utilizing geometric and relative camera pose information. The use of 2D depth prediction convolutional encoder-decoder type architecture allows this to be fast. The network is augmented with the use of metadata for the cost volume allowing depth predictions to be highly accurate. Figure 2 describes the pipeline of depth prediction for SimpleRecon.

Access to additional information like geometric and relative camera pose information that is present in the metadata aids its performance. This augmentation gives the context between images to the network. The method does not make use of complex 4D cost volume reductions, temporal fusion, or Gaussian processes, focusing instead on high-quality multi-view depth prediction. This leads to competitive 3D reconstruction quality while being fast, and suitable for real-time, low-memory environments.

3.2. 3-D Reconstruction

After we obtain the depth map from SimpleRecon, depth merging and surface reconstruction are performed using simple, off-the-shelf depth fusion techniques. The highly accurate depth predictions aid in the task. This uses Truncated Signed Distance Functions to integrate the depth maps. The method uses an off-the-shelf method called Infinitam[13]. This part of the process combines the individually predicted depths from different views into a coherent 3D model. It capitalizes on the accuracy of the predicted depths, ensuring that the final 3D reconstruction effectively represents the actual scene. This step is crucial for translating the depth information into a usable 3D format.

3.3. Obtaining point cloud

Converting mesh data to a point cloud using the vertex sampling method in Open3D [22] involves extracting the vertices from the mesh data and using them as points for the point cloud. This method is straightforward in Open3D [22], where you first load or define your mesh and then directly create a point cloud from the vertices of this mesh. This process essentially samples points at the vertices of the mesh, representing the shape and structure of the original mesh but in the point cloud format. We later did Poisson disk sampling of points directly from the Mesh to improve the density of points sampled from the mesh and ensure it

was random.

3.4. Point cloud segmentation

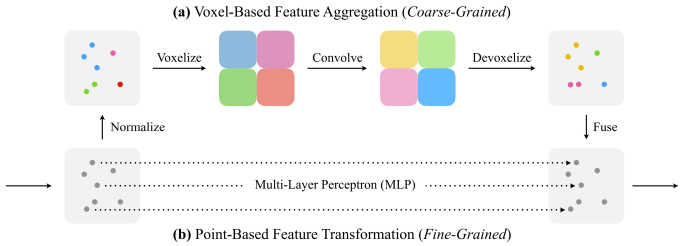


Figure 3. PVCNN: voxel and point cloud feature aggregation [10]

The Point-Voxel CNN (PVCNN) [10] makes use of high-level point cloud features as well as low-level voxel features generated from the point cloud. The process of voxel feature generation and point cloud feature extraction are split into branches as shown in figure 3.

To obtain voxel-based features from the point cloud, the following steps are done:

- The scale of point clouds is normalized within a unit sphere ensuring point features remain as such and for uniformity of values on a scale.
- The normalized point clouds get converted into voxel grids by averaging features within each voxel, based on their coordinates.
- The process of feature aggregation happens by 3D volumetric convolutions.
- To fuse the voxel feature with the point cloud features, it is devoxelized by the use of trilinear interpolation, keeping features distinct from each other.

This is the pipeline of voxel-based feature aggregation.

To capture better details, there is the task of working directly on each point to extract their features using a Multi-Layer Perceptron (MLP). The use of a simple MLP does not affect its effectiveness in producing unique features. This and the fusion of voxel features enhance the overall features obtained, the coarse voxel features with the fine point features, making them supplement each other.

4. Experiments and Results

This section lists the experiments we have done for our proposed method. We first establish a baseline performance of different configurations of SimpleRecon.Then we compare different PointNet-based Segmentation models for their accuracy. We visualize outputs from SimpleRecon and PVCNN. Finally, we demonstrate the outputs of our proposed pipeline that integrates these methods. We explain the results of the integrated pipeline in the next section.

4.1. Datasets

We use pre-trained model weights provided by the authors of SimpleRecon [18] as the hardware requirements of training from scratch were not possible. The authors trained the model on Scannet [2], which is a very large dataset, almost panning 1.2Tb in size. As a result, we used the Microsoft research RGB-D Dataset 7-Scenes [5] which consists of 7 scenes of which we used chess and office rooms. The authors of SimpleRecon demonstrated the generalization capability on the 7-Scenes dataset and we were able to confirm the same. The PointNet segmentation models were trained on S3DIS due to a lack of segmentation labels for the 7 scenes dataset.

Our combined pipeline utilizes the 7 scenes dataset for the input MVS images as well as for segmentation by PVCNN as we could not find a suitable dataset that contained point cloud segmentation Labels along with the suitable TSDF volume ground truth needed for training and evaluation by SimpleRecon for indoor environments. The possibility of converting the TSDF volumes ground truths into meshes or some similar 3D representation to manually annotate the ground truths was considered but could not be done due to the possibility of inaccurate labels and the time it would take. Similarly, the process of converting a 3D representation of ground truth in a TSDF volume and passing it as the ground truth is also a time-consuming process that has high possibility of errors in the output.

4.2. SimpleRecon

Combination Used	**RMSE**	**SQ-Rel**	**Avg Inference Time**
Dot Product only with Dense Frames	0.1799	0.02	74.86
Dot Product only with Normal Frames	0.1720	0.0164	116.519
Metadata Enabled with Dense Frames	0.1769	0.0181	133.41
Metadata Enabled with Normal Frames	0.1676	0.0153	186.41

Table 1. Model Parameter and Frame Combinations. Note only RMSE and SQ Rel Error are listed for comparison

We tested SimpleRecon on the 7 Scenes dataset. We tested four different configurations that used different sets of frames from the collection of input MVS images and with/without the addition of metadata into the model. Table 1 shows the results. The default frame combination uses keyframes in a scan that are selected by DeepMVS's [3]

heuristic as done by [18]. A dense keyframe combination makes use of every available possible keyframe in a scene.

As seen, the combination of using metadata along with default frames provides the lowest error but takes the most time. The difference in error between the combinations is low, proving that the method, regardless of keyframes and metadata being used for depth predictions and mesh reconstruction can perform well. We use the recommended combination of the model with metadata enabled and specific keyframes being skipped for the best performance.

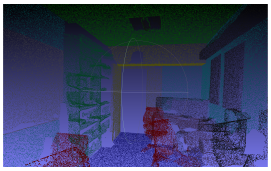
4.3. Point segmentation comparison

Model	Average accuracy	Number of epochs
Pointnet	84.93	20
Pointnet ++	85.22	20
Voxel+Point Cloud Segmentation	86.47	20
Voxel+Point Cloud Segmentation++	87.48	20

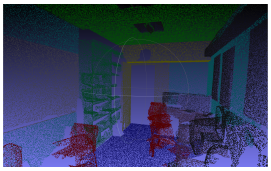
Table 2. Point segmentation accuracy comparison

We did a comparison of various point net-based point segmentation methods to benchmark their performance and demonstrate how PVCNN has good accuracy. All of the point clouds were tested and trained on S3DIS as mentioned earlier. We trained them for 20 epochs and tested them on the above-mentioned dataset.

Table 2 shows that all segmentation methods perform moderately well with PVCNN++ taking a slight lead. PVCNN++ is an extension of PCVNN proposed by the authors [10] which improves the performance of PVCNN slightly.



(a) Segmentation ground truth output



(b) Segmentation Prediction outputs

Figure 4. PVCNN Segmentation

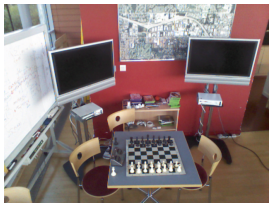
4.4. Results

In the results depicted in Figure 5, it is evident that the described pipeline performs optimally up to the point where the generated dense point cloud is passed into the PVCNN segmentation model. At this juncture, the output deviates from the expected segmentation results, ideally illustrated in Figure 4b. We were not also able to measure the error,

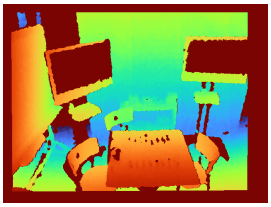
A detailed analysis reveals several potential factors contributing to our final outputs.

Our testing dataset lacks a set of ground point clouds as well as their segmentation labels. The lack of such data affected the ability to get error metrics for our predictions. This also translates to an inability to train on the same dataset due to the lack of labels. A potential solution is to manually generate the ground truth data which is labor intensive and may not be possible without the use of software designed for this application.

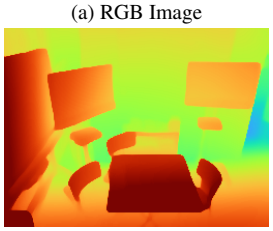
Another factor was the lack of hardware resources to undergo zero-shot learning that could have potentially helped in generalization and probably improved the segmentation output. It also resulted in an inability to train our models from scratch. The use of alternative methods for point cloud segmentation like [11] that make use of multi-model data for semantic segmentation which could work for the segmentation of new unseen classes of input data. Lai *et al*. [8] suggests the use of a transformer that has strong generalization ability and could improve our results.



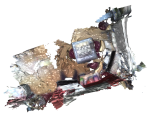
(a) RGB Image



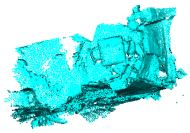
(b) Ground truth depth



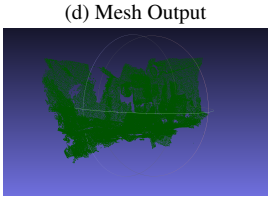
(c) Predicted depth



(d) Mesh Output



(e) Point cloud output



(f) Point cloud segmentation

Figure 5. 7 Scenes outputs

5. Analysis of Prior Pipeline

Our initially proposed framework was designed to extract point clouds from a single RGB image. We used UNET [4] to estimate the depth of the image and used that in open3D[22] to generate a point cloud. But as seen in fig-

(a) RGB Image

(b) Point cloud output from failed pipeline

Figure 6. Failed pipeline outputs

ure:6, the resultant point cloud was too sparse and cannot be used for segmentation. Pointnet [14] has shown the ability to handle sparsity of point clouds but in this case, it was not able to give a meaningful output.

Hence, we can conclude that the use of a single image to estimate the depth to generate a proper point cloud without the use of a deep learning method was not successful. As a result, we have used MVS images that that a better output and can be used for segmentation.

6. Conclusion

In this research paper, we investigated the utilization of images as input to derive meshes, which in turn can be employed for the creation of 3D indoor scene point cloud representations and schematic segmentation. Our primary objective was to leverage the efficiency and accuracy of SimpleRecon and PVCNN to generate a simplified and efficient schematic segmentation of these scenes.

While we were successful in demonstrating the viability of our proposed method and were able to visualize our outcomes, it's important to note that the performance fell short of our ideal standards. Nonetheless, the insights gained from these results are invaluable and provide a distinctive perspective within the field of 3D mapping.

Looking ahead, we identify several avenues for refinement. First, we propose the adoption of more advanced mapping techniques to produce higher-quality point clouds from depth maps, potentially yielding more accurate reconstructions. Additionally, we suggest a shift towards using a model for estimating camera poses from video sequences as opposed to relying solely on Multi-View images. This alteration has the potential to significantly enhance the performance of our 3D reconstruction efforts. Lastly, we consider the benefits of cross-data set training for the Pointnet segmentation model to improve its generalization capabilities. These adjustments hold promise for substantially boosting the performance of our method.

In summary, this project serves as a foundation for future explorations in the field, providing valuable insights and opening up possibilities for further advancements in 3D mapping techniques.

7. Individual Contributions

All members of our team contributed equally to this project.

Nishant Pandey - Report, Coding, and Literature Survey

Jayasuriya Suresh - Report, Debugging Code, and Literature Survey

Rishikesh Jadhav - Report, Debugging Code and Literature Survey

8. Credits

We would like to thank the authors of Simple Recon and PVCNN for open-sourcing their code and Stanford, Microsoft and Scannet for their excellent datasets. We would also like to thank the Professor and Teaching Assistants for their guidance in this project and the opportunity to research this topic.

References

[1] Jeremy Castagno and Ella Atkins. Polylidar3d – fast polygon extraction from 3d data, 2020. 1, 2

[2] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. 4

[3] Arda Düzçeker, Silvano Galliani, Christoph Vogel, Pablo Speciale, Mihai Dusmanu, and Marc Pollefeys. Deepvideomvs: Multi-view stereo on video with recurrent spatio-temporal fusion, 2021. 4

[4] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network, 2014. 5

[5] Ben Glocker, Shahram Izadi, Jamie Shotton, and Antonio Criminisi. Real-time rgb-d camera relocalization. In *International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2013. 4

[6] Samer Karam, Francesco Nex, Bhanu Teja Chidura, and Norman Kerle. Microdrone-based indoor mapping with graph slam. *Drones*, 6(11), 2022. 2

[7] Georg Krispel, Michael Opitz, Georg Waltner, Horst Possegger, and Horst Bischof. Fuseseg: Lidar point cloud segmentation fusing multi-modal data. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2020. 2

[8] Xin Lai, Jianhui Liu, Li Jiang, Liwei Wang, Hengshuang Zhao, Shu Liu, Xiaojuan Qi, and Jiaya Jia. Stratified transformer for 3d point cloud segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8500–8509, 2022. 2, 5

[9] Minghua Liu, Yinhao Zhu, Hong Cai, Shizhong Han, Zhan Ling, Fatih Porikli, and Hao Su. Partslip: Low-shot part segmentation for 3d point clouds via pretrained image-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21736–21746, 2023. 2

[10] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2019. 1, 2, 3, 4, 5

[11] Yuhang Lu, Qi Jiang, Runnan Chen, Yuenan Hou, Xinge Zhu, and Yuexin Ma. See more and know more: Zero-shot point cloud segmentation via multi-modal visual data, 2023. 5

[12] Sagar Manglani. Real-time vision-based navigation for a robot in an indoor environment, 2023. 2

[13] Victor Adrian Prisacariu, Olaf Kähler, Stuart Golodetz, Michael Sapienza, Tommaso Cavallari, Philip H S Torr, and David W Murray. Infinitam v3: A framework for large-scale 3d reconstruction with loop closure, 2017. 3

[14] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation, 2017. 2, 6

[15] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space, 2017. 2

[16] Damien Robert, Bruno Vallet, and Loic Landrieu. Learning multi-view aggregation in the wild for large-scale 3d semantic segmentation, 2022. 2

[17] Erik Sandström, Yue Li, Luc Van Gool, and Martin R. Oswald. Point-slam: Dense neural point cloud-based slam, 2023. 1, 2

[18] Mohamed Sayed, John Gibson, Jamie Watson, Victor Prisacariu, Michael Firman, and Clément Godard. Simplerecon: 3d reconstruction without 3d convolutions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 1, 2, 3, 4, 5

[19] Daniel Seichter, Patrick Langer, Tim Wengefeld, Benjamin Lewandowski, Dominik Hoechemer, and Horst-Michael Gross. Efficient and robust semantic mapping for indoor environments, 2022. 1, 2

[20] Zhifeng Teng, Jiaming Zhang, Kailun Yang, Kunyu Peng, Hao Shi, Simon Reiß, Ke Cao, and Rainer Stiefelhagen. 360bev: Panoramic semantic mapping for indoor bird's-eye view, 2023. 1, 2

[21] Chen Yang, Qi Chen, Yaoyao Yang, Jingyu Zhang, Minshun Wu, and Kuizhi Mei. Sdf-slam: A deep learning based highly accurate slam using monocular camera aiming at indoor map reconstruction with semantic and depth fusion. *IEEE Access*, 10:10259–10272, 2022. 2

[22] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3d: A modern library for 3d data processing, 2018. 3, 5

[23] Nicky Zimmerman, Matteo Sodano, Elias Marks, Jens Behley, and Cyrill Stachniss. Constructing metric-semantic maps using floor plan priors for long-term indoor localization, 2023. 2